A SYSML REQUIREMENTS MODEL FOR THE 1992 ACC ROBUST CONTROL BENCHMARK

Fernando Valles-Barajas

Department of Information Technology, Faculty of Engineering, Universidad Regiomontana
15 de Mayo 567 Pte., C.P. 64000 Colonia Centro, Monterrey, Nuevo León, México
e-mail: fernando.valles@acm.org, fernando.valles@ieee.org

Abstract. In this paper a conceptual design for the software of control systems is presented. The design was made using requirements tables and requirements diagrams of SysML, which is a modeling language for modeling systems. The resulting design can be used to estimate the required effort to build software for control systems as well as to document and complement the textual requirements of control systems. The usefulness of this proposal is shown by making a conceptual design for the 1992 ACC robust control benchmark.

Keywords: PSP, Software Design, Control Systems, SysML, Requirements Diagrams.

1. Introduction

The Personal Software Process[1] (PSP) is a process that helps software engineers in producing high quality software [4]. In this software process a software engineer needs to estimate the necessary effort to make a software product based on a requirements document. For this estimation, PSP recommends making a model which captures the essential characteristics of the system to be developed; this model is called in PSP conceptual design. Once the estimation of the necessary effort is made, the conceptual design is detailed later using the four PSP templates (operational, functional, sate and logic templates) which model four different views of a system.

SysML is a modeling language that has several diagrams and is used to model systems [6]. One of the SysML diagrams is the requirements diagram, which is used to model, document, specify, and analyze the requirements of a system.

Control systems are composed of hardware (sensors, computers, actuators) and software [5]. The latter is where control laws (for example PID, RST or GPC) are specified.

Motivation of the paper: the author of this paper proposes to make a conceptual design for the software of control systems based on a requirements document using requirements tables and SysML requirements diagrams. The benefits of the proposal are:

- This design can be used to estimate the necessary effort to build software for control systems.
- This design will later serve to build a detailed design for the software of control systems using the four PSP templates.
- A graphical model where requirements are specified is added to the textual documentation of control systems.
- A bridge between typical requirements tools and tools that model the system elements is established. Using this bridge, when a requirement changes the model elements that suffer a change will easily be detected.

Related works: In [10] a requirements process for the software of control system is presented. This process is based mainly in the Rational Unified Process (RUP).
In [2] the specification and analysis of the requirements for an avionic control system are given. This specification is based on formal methods. In particular the PVS language is used in the specification and analysis phase.
The importance of the specification of control systems requirements is shown by taking a look at the special issue on the modeling and analysis of the requirements for a light control system [8].

Structure of the paper: In Section 2 SysML requirements diagrams are explained. Section 3

[1] © Software Engineering Institute

contains the requirements in textual form for the 1992 ACC robust control benchmark. In section 4 a conceptual design of the control system explained in sec. 3 is given using the SysML notation. The last section presents some concluding remarks.

2. SysML

SysML is a modeling language that is based on UML and serves to model systems. It takes some of the UML diagrams (sequence, state machine, use case and package diagrams), modifies some of them (activity, block definition and internal block diagrams) and introduces new diagrams (requirements and parametric diagrams).

2.1. The requirements diagram

A requirement is a capability or condition that a system must satisfy. In SysML the requirements are represented as model elements; being more specific, the requirements in a requirements diagram are represented as abstract classes without operations. A requirement has properties to specify its unique *id* and its description; the latter property is called "*text*".
A requirement is represented as a stereotyped class using the stereotype «requirement» with a unique id and a textual description.

The relationships between requirements in a requirements diagram are: containment, which is represented using a cross hair, and the stereotyped dependencies «deriveReq», «copy» and «refine». There are other relations between requirements and model elements. For example, the «satisfy» relationship is used to specify a model element that satisfies some particular requirement. The «verify» relationship is used to assign a test case to a model element.
The containment relationship is used to decompose a requirement into sub requirements; so a complex requirement req_1 can be decomposed in, for instance, requirements req_2 and req_3. The containment relationship is useful to deal better with the implementation of complex requirements. Requirements related to the containment relationship form a hierarchy. This relationship is represented by a small circle with a cross collocated near the complex requirement.
The «copy» relationship is used to reuse requirements in different models. It is important to mention that the «copy» relationship is necessary, because in SysML a model element can not be drawn in two different diagrams.

The «deriveReq» relationship can be used to specify when one requirement is a derivation from another requirement. This relationship expresses the relation between a particular requirement and the requirements that support this requirement.
The «refine» relationship can be used to specify an element that refines a requirement; for example a use case or a sequence diagram can be used to refine a requirement.
It is possible to define the design or implementation model which satisfies one or more requirements by using the «satisfy» relationship.
Test cases can be associated with a requirement by using the «verify» relationship. The notation for a test case is similar to the notation of a requirement; a stereotyped class with the stereotype «testCase» is used to represent a test case.
The «rationale» construct can be attached to a requirement or to a relation to support, for example, the decision that was taken to satisfy a requirement using a component.

A requirements diagram is a graphical form to represent the requirements of a system. The requirements of a system, the relationships between them and other elements can also be represented in tabular form. Table 1 shows an example of a table that specifies the requirements of a system. This table contains an *id*, a *name* and a *description* of the requirements, but this table may also contain the relationships between them.

Table 1. Tabular form for the requirements

id	name	description
req_1	requirement name	The system shall..

3. A study case

Fig. 1 presents a typical configuration for a control system. In this figure $y(k)$ is the output

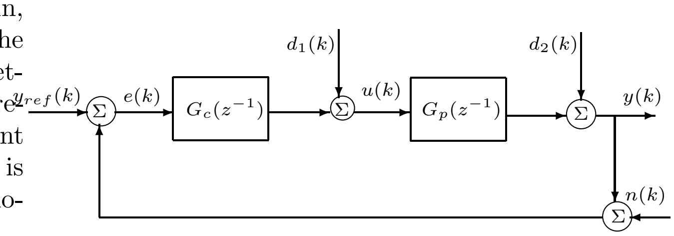

Figure 1. Typical configuration for a control system

variable, $u(k)$ is the input variable, $e(k)$ is the feedback error, $y_{ref}(k)$ is the reference, $d_1(k)$, $d_2(k)$ and $n(k)$ respectively are a disturbance on the input, a disturbance on the output and noise affecting the measured variable. The process being controlled is represented by the transfer function $G_p(z^{-1})$ which is defined by:

$$G_p(z^{-1}) = \frac{z^{-d}B(z^{-1})}{A(z^{-1})} \quad (1)$$

where:

$$A(q^{-1}) \;=\; 1 + q^{-1}A^*(q^{-1}) = 1 + \sum_{1}^{n_A} a_i q^{-i}$$

$$B(q^{-1}) \;=\; q^{-1}B^*(q^{-1}) = \sum_{1}^{n_B} b_i q^{-i}$$

and the delay d, is a multiple integer of the sampling period T_s. $G_c(z^{-1})$ is the controller.

Suppose that it is required to construct a control system, based on the configuration of fig. 1, that fulfills the following requirements:

Textual form of the requirements:

1. It is required that the control law considers the energy used by the final element; this element is not shown in fig 1 but this energy will be represented by the variable $u(k)$.
2. The system shall be able to specify the response of $G_c(z^{-1})$ to a specific input applied to $y_{ref}(k)$ (for instance a step input) and to the disturbance affecting $y(k)$.
3. There is no restriction on the order of the transfer function of $G_p(z^{-1})$ and on the size of the time delay.
4. The system shall deal with processes which $B^*(z^{-1})$ has unstable zeros.
5. From previous experiments, it is known that the behavior of $G_p(z^{-1})$ may change depending on the zone of operation. Because of this, the system shall deal with parametric variations of $G_p(z^{-1})$.
6. The system shall be insensitive to uncertainty, noise and disturbances affecting $y(k)$ and $u(k)$.
7. The system shall manage any fault that could affect the control system.

To be more specific, let us analyze the requirements for the ACC 1992 robust control benchmark [9]. This system is composed of two masses connected to a single spring (see fig. 2). The output variable for this system, y, is the position of m_2, represented by x_2, u is the control input acting on m_1, k is the spring constant; also there are two disturbances, d_1 and d_2 acting respectively on m_1 and m_2. x_3 and x_4 represent respectively the velocity of x_1 and x_2.

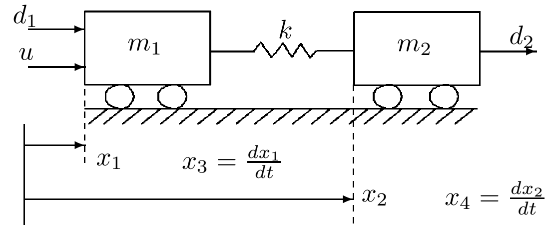

Figure 2. The two-mass-spring system

Assuming that $d_1 = 0$ the transfer function between $y(k)$ and $u(k)$ is

$$G_{uy}(s) = \frac{\frac{k}{m_1 m_2}}{s^2\left[s^2 + \frac{k(m_1+m_2)}{m_1 m_2}\right]} \quad (2)$$

The transfer function between $y(k)$ and $d_2(k)$ is

$$G_{d_2 y}(s) = \frac{\left(\frac{1}{m_2}\right)\left(s^2 + \frac{k}{m_1}\right)}{s^2\left[s^2 + \frac{k(m_1+m_2)}{m_1 m_2}\right]} \quad (3)$$

The main requirement for this system is to design a feedback compensator for the 4th-order spring-mass system whose parameters are bounded but unknown. When the 1992 ACC robust control benchmark was presented, four scenarios were proposed. The following constraints correspond to scenario 4:

1. The effort of the controller is limited to $|u(k)|$ less than or equal to 1 (requirement 1 of the previous list).
2. Settling time t_s and overshoot M_p are both minimized. t_s is achieved when $y(k)$ is bounded by ± 0.1 units (requirement 2 of the previous list).
3. The system is robust with respect to the perturbations d_1 and d_2 in m_1, m_2 and parametric variations of k. The nominal parameters of the process are: $m_1 = m_2 = k = 1$ (requirement 3 of the previous list).

Requirement 4 of the previous list is needed in the scenario one of the 1992 ACC robust control benchmark. In the rest of the paper it will be assumed that this system has been discretized.

4. The model

This section presents a conceptual design for the requirements document of the 1992 ACC robust control benchmark based on SysML requirements diagrams and requirements tables.

Table 2. Requirements for $G_c(z^{-1})$

id	name	text	relatesTo	relatesHow
req_1	$u(k)$	The energy of $u(k)$ shall be regulated by $G_c(z^{-1})$	req_8	«satisfy»
req_2	$y(k)$ and $d_2(k)$	The specification of the response of $y(k)$ and $d_2(k)$ shall be allowed	req_8	«satisfy»
req_3	$G_p(z^{-1})$ and d	There is no restriction in the order of $B(z^{-1})$ and $A(z^{-1})$ and the size of d	req_8	«satisfy»
req_4	$B^*(z^{-1})$	The system shall deal with processes which $B^*(z^{-1})$ has unstable zeros	req_8	«satisfy»
req_8	RST $G_c(z^{-1})$		$req_1 - req_4$	«satisfy»
			req_9	«deriveReq»
req_9	$B(z^{-1})$ and $A(z^{-1})$	$B(z^{-1})$ and $A(z^{-1})$ must be coprime polynomials	req_8	«deriveReq»

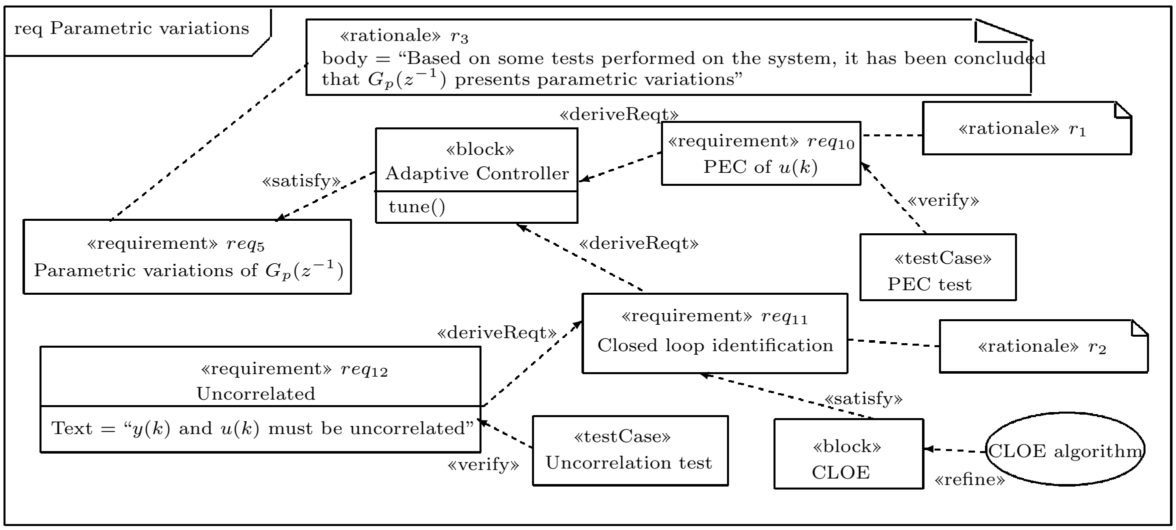

Figure 3. Requirements diagram for parametric variations

Table 2 contains the requirements related to $G_c(z^{-1})$. As can be seen in this table, an RST controller satisfies the requirements 1-4 of the requirements specification of the previous section. It can be possible by using an RST controller, to specify the response of the system when a step input is applied in the reference $y_{ref}(k)$ or when the output disturbance $d_2(k)$ occurs. An RST controller has the ability to control a process with a transfer function of any order. Controllers like PID lack of this feature [1]. Another disadvantage of a PID controller is that it has some constraints in the relationship between the dead time and the constant time of the process; an RST controller does not have this restriction. The controller design method "tracking and regulation with weighted input", which is used in the tuning of RST controllers, can deal with processes that the polynomial $B^*(z^{-1})$ has unstable zeros. In table 2 a derived requirement from the requirement *RST* $G_c(z^{-1})$ was defined, this is a relationship between $A(z^{-1})$ and $B(z^{-1})$ which states that these polynomials must be coprime[2]. In other words, if ones uses an RST controller $A(z^{-1})$ and $B(z^{-1})$ must be coprime polynomials.

Fig. 3 shows a requirements diagram that focuses on requirement 5 which was specified in the previous section. An adaptive controller can deal with time-varying process [1]; this is the reason the block *Adaptive Controller* was chosen to satisfy the requirement *parametric variations* of $G_p(z^{-1})$. This kind of controller was considered to satisfy one of the requirements for the first scenario of the ACC robust control benchmark: the parameter k is bounded but variable. The rationale r_3 justifies the importance of requirement 5. Two requirements are derived from the block

[2] polynomials $A(z^{-1})$ and $B(z^{-1})$ are coprime if there is not a common factor between them.

Adaptive Controller: *PEC of $u(k)$* and *Closed loop identification*. An adaptive controller identifies the model of the process based on measurements of the input $u(k)$ and output $y(k)$ of the process. To get a good model, the content of information of the signal $u(k)$ must be good enough to find the parameters of $G_p(z^{-1})$. Formally speaking the signal $u(k)$ must fulfill the Persistent of Excitation Condition (PEC). In what follows the rationale for the PEC is explained; this construct is labeled as r_1 in the fig. 3.

4.1. Rationale for PEC, r_1

Let us represent equation 1 in the difference equation:

$$y(k+1) = -\sum_{i=1}^{n_A} a_i y(k-i) + \sum_{i=1}^{n_B} b_i u(k-d-i+1) \tag{4}$$

It is desirable to identify the $(n_A + n_B)$ parameters of the model. According to [5], a condition that must be fulfilled in order to achieve this task is that the input variable $u(k)$ has p-sinusoids components of different frequency $(u(k) = \sum_{i=1}^{p} sin(\omega_i T_s k))$. If $(n_A + n_B)$ is even, then p must be greater than or equal to $\frac{n_A+n_B}{2}$ and if $(n_A + n_B)$ is odd, then p must be greater than or equal to $\frac{n_A+n_B+1}{2}$. This condition is known as the Persistent Excitation Condition and is related to the information content of the input variable $u(k)$. If the PEC is not fulfilled, the parameters calculated will be either outside the valid zone[3] or inadmissible.

4.2. Rationale for closed loop identification, r_2

As can be seen in fig. 3 the requirement *parametric variations of $G_p(z^{-1})$* was satisfied by the block *adaptive controller*. This kind of controller performs identification[4] in closed loop in order to detect any change in the dynamics of the process and then to update the model $G_p(z^{-1})$; this is the reason the requirement *closed loop identification* was defined as a derived requirement of the block *Adaptive Controller*. Unfortunately, the performance of an identification in closed loop is decreased when the input of the process $u(k)$, the disturbance $d_2(k)$ and noise $n(k)$ (see fig. 1) are correlated and because of this the parameters identified are biased.

In what follows, a formal explanation of the fundamental problem of closed-loop identification is given. This explanation is based on [3].

Suppose that the process represented by equation 4 only depends on values of $u(k)$:

$$\begin{aligned} y(k) &= B(z^{-1})u(k) + e(k) \\ &= b_1 u(k-1) + \cdots + b_n u(k-n) + e(k) \end{aligned} \tag{5}$$

where $e(k)$ represents disturbance $d_2(k)$ and noise $n(k)$ acting on the output $y(k)$. This model can be represented in the following way:

$$y(k) = \phi^T(k)\theta + e(k) \tag{6}$$

where $\theta = [b_1 \dots b_n]^T$ is the parameter vector of $G_p(z^{-1})$ and

$$\phi^T(k) = [u(k-1) \dots u(k-n)]^T \tag{7}$$

is the vector of measurements ($y(k)$ and $u(k)$), which in this case is composed only by values of $u(k)$.
Using the least square algorithm to find an estimation for the vector θ given N data points, the next equation is obtained:

$$\begin{aligned} \hat{\theta}(N) &= \left[\frac{1}{N}\sum_{k=1}^{N}\phi(k)\phi^T(k)\right]^{-1}\frac{1}{N}\sum_{k=1}^{N}\phi(k)y(k) \\ &= \theta + \left[\frac{1}{N}\sum_{k=1}^{N}\phi(k)\phi^T(k)\right]^{-1}\frac{1}{N}\sum_{k=1}^{N}\phi(k)e(k) \end{aligned} \tag{8}$$

Under mild conditions on the data set we then have that $\hat{\theta}(N)$ approaches to θ^* with probability 1 where

$$\begin{aligned} \theta^* &= \lim_{N\to\infty} E\hat{\theta}(N) \\ &= \theta + \left[\bar{E}\phi(k)\phi^T(k)\right]^{-1}\bar{E}\phi(k)e(k) \end{aligned} \tag{9}$$

where Ex is the mathematical expectation of the random vector x and the "total expectation" operator $\bar{E}$ is defined as

$$\bar{E}x(k) = \lim_{N\to\infty}\frac{1}{N}\sum_{k=1}^{N}Ex(k) \tag{10}$$

The least square algorithm is consistent if

$$\bar{E}\phi(k)e(k) \equiv 0 \tag{11}$$

but this is not achieved because $u(k)$ and $e(k)$ are correlated due to the fact that the identification

[3] the set of parameters that corresponds to the process is called valid zone

[4] identification is the procedure by which a model of the process is obtained.

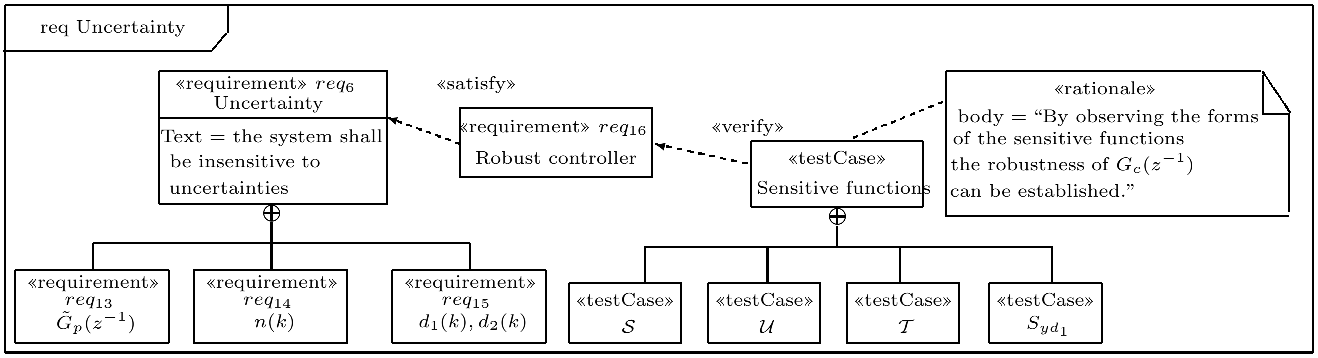

Figure 4. Requirements diagram for the uncertainty in the control system

procedure is carried out in closed loop. This is the reason the derived requirement *Uncorrelated* was defined in fig. 3.

It is important to mention that the algorithm used for identification in an adaptive controller is a recursive version of the Least Square Algorithm and it can be shown that the same problem will be presented using this algorithm. The non recursive version of the Least Square Algorithm was used to explain the problem because it is easier to give an explanation of this algorithm.

In fig. 3, the *closed loop identification* requirement is satisfied by the CLOE block which is an algorithm that performs identification in closed loop [5]. This algorithm is then refined by the use case *CLOE algorithm*. Two *testCase* blocks were defined in fig. 3: *PEC test* and *Uncorrelation test*. These blocks will ensure the proper work of the adaptive controller.

Fig. 4 shows a requirements diagram that focuses in the uncertainty in the control system (requirement 6). To efficiently manage the uncertainty in the system, this requirement was divided into three parts by using the composition relation: *parametric variations of* $G_p(z^{-1})$, *noise* $n(k)$ and *external disturbances* $d_1(k)$ *and* $d_2(k)$. In figure 4, the *uncertainty* requirement is satisfied by a robust controller. To verify the work of this controller a *testCase* called *sensitive functions* is defined. The sensitive functions reflect how the system is affected by disturbances $d_1(k)$, $d_2(k)$ and noise $n(k)$. By drawing these functions, a control engineer can determine how robust the system is with respect to these external signals. The output sensitive function $\mathcal{S}$ models the relationship between $y(k)$ and $d_2(k)$. A measurement of how the output disturbance $d_2(k)$ affects the input $u(k)$ is given by the input sensitive function $\mathcal{U}$. The relationship between noise $n(k)$ and the output $y(k)$ is known as the complementary sensitive function $\mathcal{T}$. The last sensitive function, $S_{yd_1}(z^{-1})$, establishes a relationship between the disturbance $d_1(k)$ and the output $y(k)$.

In fig. 4 the composition relationship was used to model the relation between the test case *sensitive functions* and the test cases for each of the sensitive functions. It is important to mention that this relationship between test cases is not defined in SysML but the author of this paper believes that a complex test case can be managed more efficiently by using this relationship.

Requirement 7, which reflexes the capacity of the control system to deal with faults, can be satisfied by using a FDI system which is composed of an isolation and detection block [7].

5. Conclusions

In this paper a novel approach to make a conceptual design for the software of control systems has been presented. This approach is based on the notation used in SysML. An example (the 1992 robust control benchmark) to demonstrate the applicability of the approach was given. By using this approach, control engineers can estimate the necessary effort to build the software for a control system.

References

[1] **K. Aström and K. J. Witternmark.** Adaptive Control. Addison-Wesley, USA, 2nd edition, 1995.

[2] **B. Dutrerte and V. Stavridou.** Formal requirements analysis of an avionic control system. *IEEE Transaction on software engineering*, 23(5), 1997.

[3] **U. Forssell and L. Ljung.** Closed-loop identification revisited. *Automatica*, 35(7), 1215-1241. 1999.

[4] **W. S. Humphrey.** PSP : A Self-Improvement Process for Software Engineers. Addison-Wesley, USA, 2005.

[5] **I. D. Landau and G. Zito.** Digital Control Systems: Design, Identification and Implementation. Springer-Verlag London Ltd, London, 2006.

[6] **OMG.** Systems Modeling Language (SysML) Specification. Object Management Group, 2006.

[7] **R. J. Patton, P. M. Frank, and R. N. Clark.** Issues of Fault Diagnosis for Dynamic Systems. Springer-Verlag, 2007.

[8] **S. Queins, G. Zimmermann, M. Becker, M. Kronenburg, C. Peper, R. Merz, and J. Schäfer.** The light control case study: Problem description. *Journal of Universal Computer Science*, 6(7):586–596, 2000.

[9] **W. Reinelt.** Robust control of a two-mass-spring system subject to its input constraints. *In: American Control Conference*, 2000.

[10] **F. Valles-Barajas** A requirements engineering process for control engineering software *Innovations in Systems and Software Engineering: A NASA Journal*, 3(4), 217-227. 2007.

Received February 2009.